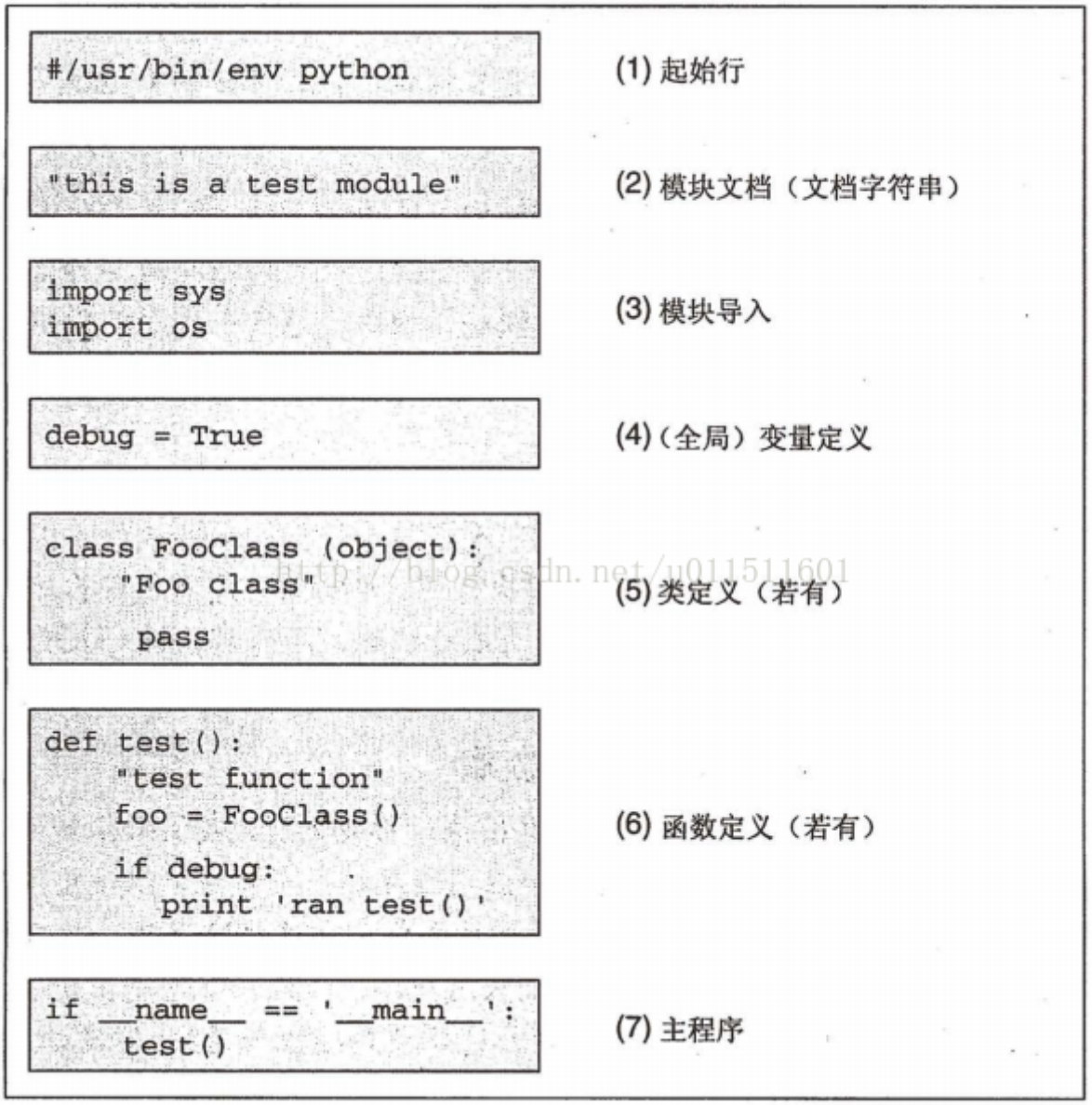
Python使用缩进对齐组织代码的执行，所有没有缩进的代码，都会在载入时自动执行。每个文件（模块）都可以任意写一些没有缩进的代码，并在载入时自动执行。为了区分 主执行代码和被调用文件，Python引入了变量：__name__。

1）当文件是被调用时，__name__的值为模块名；

2）当文件被执行时，__name__的值为 ‘__main__’。

基于此特性，为测试驱动开发提供了很好的支持，我们可以在每个模块中写上测试代码，这些测试代码仅当模块被Python直接执行时才会运行，代码和测试完美的结合在一起。

1、典型的Python文件结构：



2、Python中的__name__举例

__name__ ： 是否为主文件

```
[python]view plaincopy
#hello.py
def sayHello():
    str="hello"
print(str);
if __name__ == "__main__":
print ('This is main of module "hello.py"')
    sayHello()
```

python作为一种脚本语言，我们用python写的各个module都可以包含以上那么一个类似c中的main函数，只不过python中的这种__main__与c中有一些区别，主要体现在：

1、当单独执行该module时，比如单独执行上面的hello.py程序： python hello.py，则输出

```
This is main of module "hello.py"
hello
```

可以理解为"if __name__=="__main__":" 这一句与c中的main()函数所表述的是一致的，即作为入口；

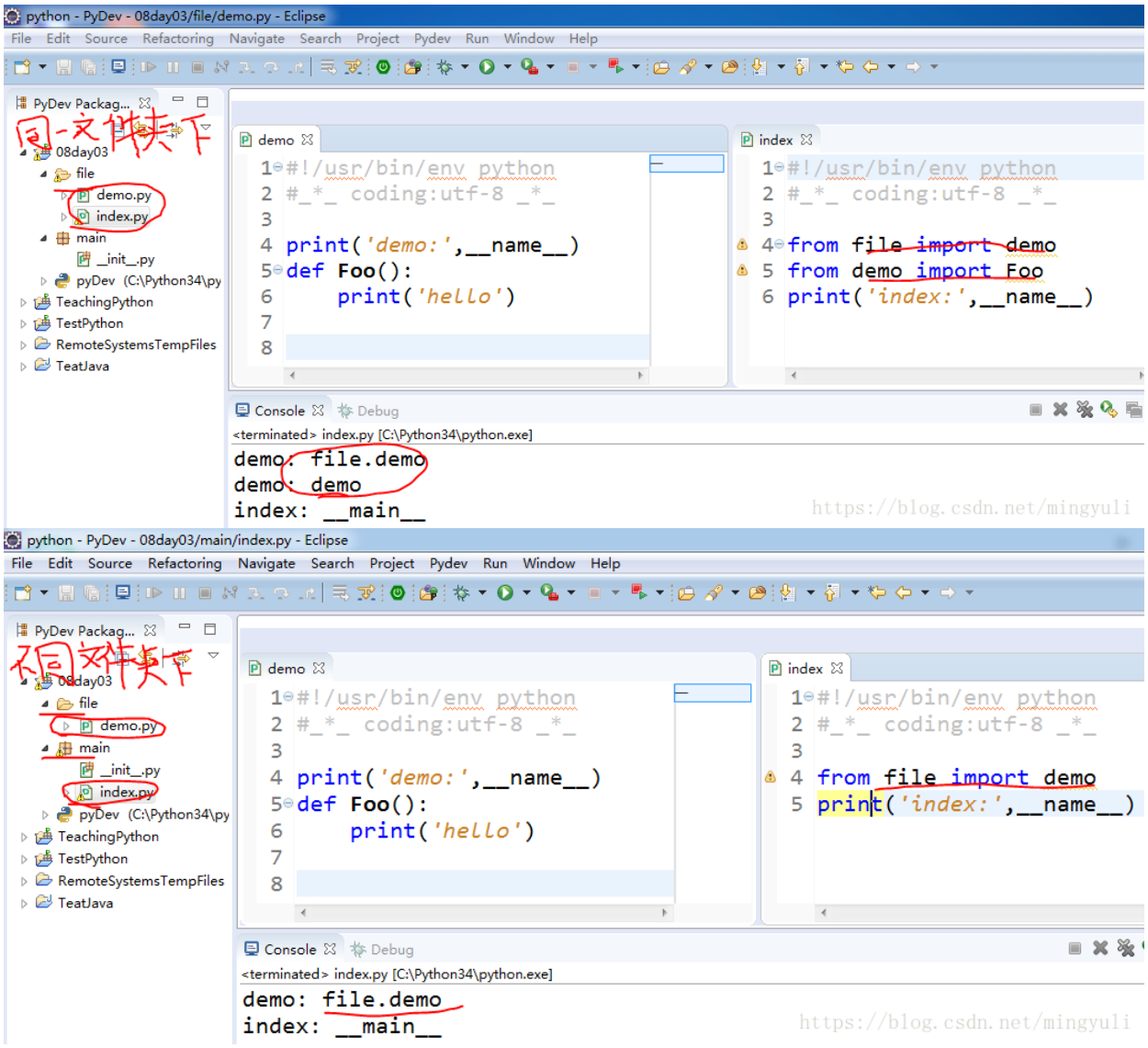
2、当该module被其它module 引入使用时，其中的"if __name__=="__main__":"所表示的Block不会被执行,这是因为此时module被其它module引用时，其__name__的值将发生变化，__name__的值将会是module的名字。比如在python shell中import hello后，查看hello.__name__：

```
>>> import hello
>>> hello.__name__
'hello'
```

3、在python中，当一个module作为整体被执行时,moduel.__name__的值是"__main__";

当一个module被其它module引用时，module.__name__将是module自己的名字；当然一个module被其它module引用时，其本身并不需要一个可执行的入口main了。

3、python中的导入举例



注意：单独执行任何一个文件时，输出都是 ： __main__

通过文件被导入执行时，输出是： 模块名.文件名

4、__file__ 、__doc__的用法

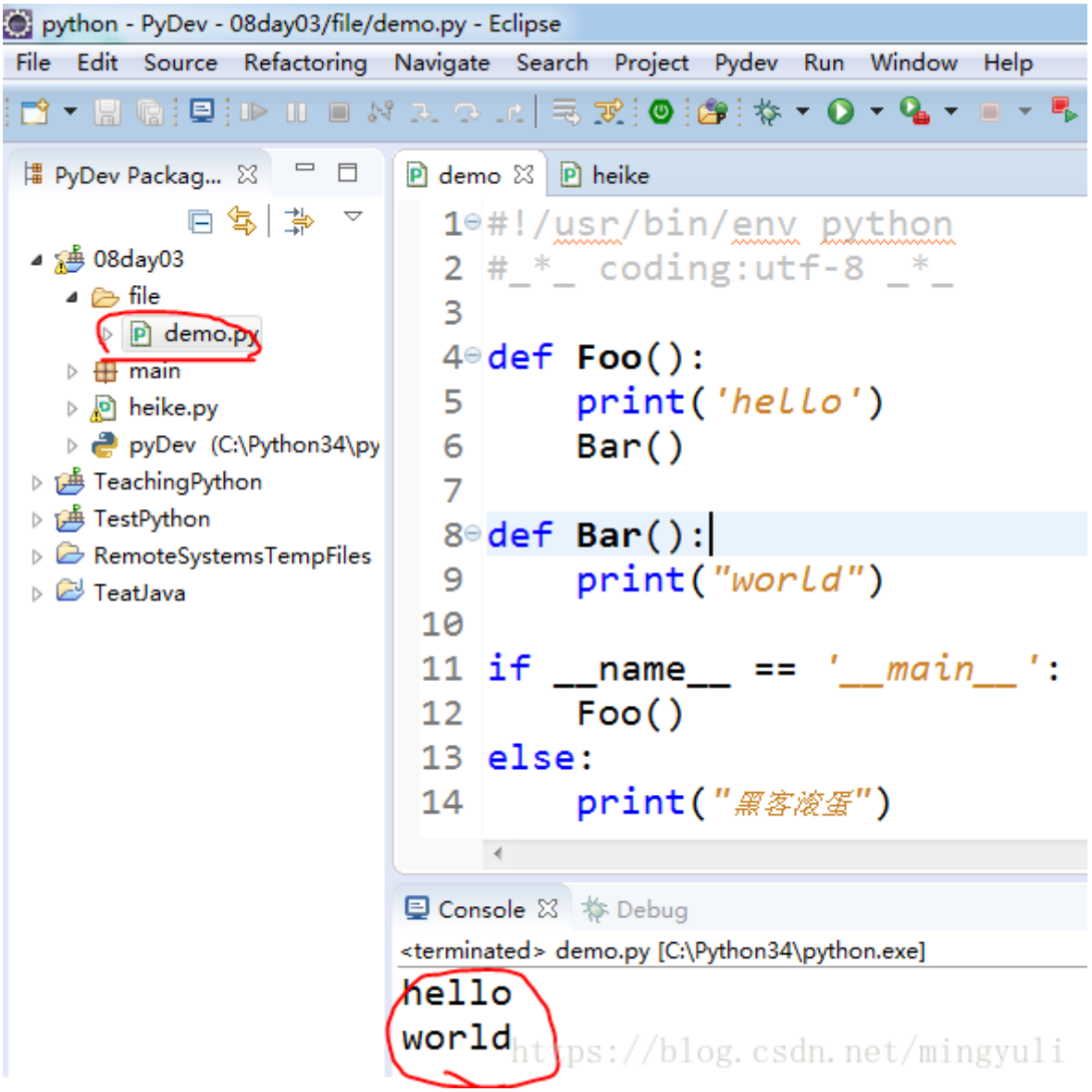
__file__ ： 当前文件路径。根据模块执行时的

__doc__ ： 当前文件描述

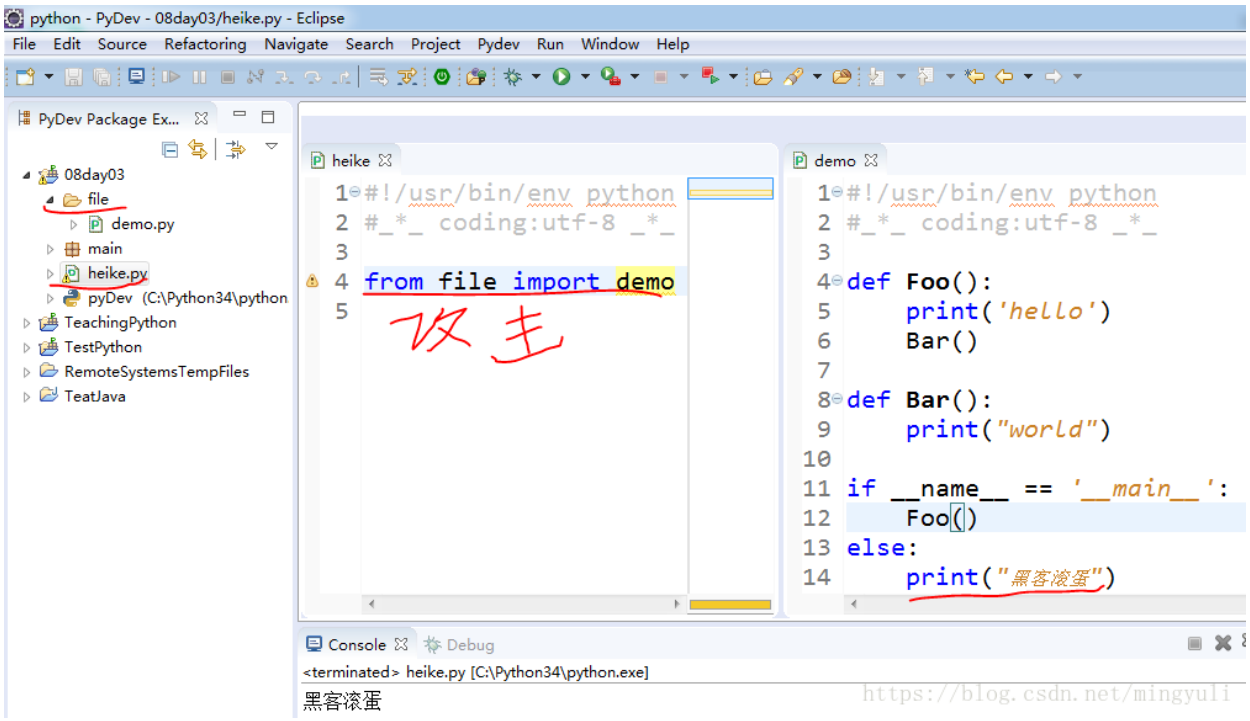
5、__name__好处：避免黑客攻击

当有黑客攻击（即从外部我们的程序）时，使用__name__可防止自己的程序被黑客执行修改

(1) 、 正常执行：



（2）、黑客攻击：从外表访问我们的程序


python - PyDev - 08day03/heike.py - Eclipse
File Edit Source Refactoring Navigate Search Project Pydev Run Window Help
PyDev Package Ex...
08day03
file
demo.py
main
heike.py
pyDev (C:\Python34\python
TeachingPython
TestPython
RemoteSystemsTempFiles
TeatJava
heike
1#!/usr/bin/env python
2#_*_ coding:utf-8 _*_
3
4from file import demo
5
demo
1#!/usr/bin/env python
2#_*_ coding:utf-8 _*_
3
4def Foo():
5 print('hello')
6 Bar()
7
8def Bar():
9 print("world")
10
11if __name__ == '__main__':
12 Foo()
13else:
14 print("黑客滚蛋")
Console
Debug
<terminated> heike.py [C:\Python34\python.exe]
黑客滚蛋
https://blog.csdn.net/mingyuli